

# TP 2: Análisis de Redes Sociales

---

Octubre 27, 2022

Métodos Numéricos

## Grupo 18

Integrante	LU	Correo electrónico
Vekselman, Natán	338/21	natanvek11@gmail.com
Arienti, Federico	316/21	fa.arianti@gmail.com
Manuel Lakowsky		
Brian Kovo	1218/21	brian.ilank@gmail.com



**Facultad de Ciencias Exactas y Naturales**  
Universidad de Buenos Aires  
Ciudad Universitaria - (Pabellón I/Planta Baja)  
Intendente Güiraldes 2610 - C1428EGA  
Ciudad Autónoma de Buenos Aires - Rep. Argentina  
Tel/Fax: (++54 +11) 4576-3300  
<http://www.exactas.uba.ar>

## RESUMEN

La descomposición de matrices en autovectores y autovalores aparece en una variedad de aplicaciones donde importa caracterizar el comportamiento de un sistema: en el reconocimiento de imágenes, en el análisis de estabilidad de cuerpos rotantes, en el análisis de riesgo de mercado y en el análisis de redes —por nombrar algunas—. Desde un punto de vista geométrico, se puede considerar a los autovectores como los ‘ejes’ de una transformación lineal, en tanto representan una dirección invariante a la transformación, y a los autovalores como los factores por los que esas direcciones se comprimen, estiran o invierten.

En este trabajo propondremos una implementación en C++ de un método para el cálculo de autovalores y autovectores en matrices cuadradas. Para algunas matrices particulares, como pueden ser las matrices simétricas definidas positivas, este método nos permitirá obtener todos sus autovalores y autovectores asociados. El mismo se conoce como *el método de la potencia con deflación*.

A su vez, presentaremos dos aplicaciones concretas de los autovalores y autovectores en el análisis de redes: la medición de centralidad de autovector y corte mínimo en la red del ‘Club de Karate’ [6], y la estimación de una *ego-network* [4] de Facebook, por medio de la construcción de una matriz de similaridad.

**Palabras clave:** *método de la potencia, deflación de Hotelling, centralidad de autovector, conectividad algebráica, análisis de componentes principales.*

## CONTENIDOS

1. Método de la potencia con deflación	2
1.1. Introducción teórica	2
1.2. Implementación	3
1.3. Evaluación cuantitativa	5
2. Análisis: Club de Karate	8
2.1. Contexto	8
2.2. Centralidad de Autovector	9
2.3. Autovectores de la matriz laplaciana	10
3. Análisis: Red ‘Ego’	14
3.1. Contexto	14
3.2. Matriz de similaridad	15
3.3. Comparación con la red original	17
3.4. PCA	19
4. Conclusiones	21
5. Apéndice	22
Referencias	23

## 1. MÉTODO DE LA POTENCIA CON DEFLACIÓN

**1.1. Introducción teórica.** El método de la potencia con deflación permite aproximar un subconjunto de los autovalores y autovectores asociados a una matriz. Si la misma satisface que todos sus autovalores son no nulos y diferentes en módulo, entonces permite aproximar el conjunto entero.

MÉTODO DE LA POTENCIA. El método de la potencia, *Power method* o *Power iteration*, es una técnica iterativa para aproximar el autovector asociado al autovalor en módulo máximo de una matriz cuadrada que satisface esta característica —es decir, tenga un autovalor dominante no nulo—, a partir de la aplicación de sucesivos productos matriciales, descriptos por la siguiente relación de recurrencia:

$$(1) \quad b_{k+1} = \frac{\mathbf{A}b_k}{\|\mathbf{A}b_k\|} \quad \forall k \in \mathbf{N}_0$$

donde  $b_0$  es un vector aleatorio,  $\|b_0\| = 1$  y  $\|\cdot\|$  es una norma vectorial.

Se puede demostrar [2] que, bajo las condiciones descriptas, si  $k$  es par y  $b_0$  no es ortogonal al autovector asociado al autovalor en módulo dominante de  $\mathbf{A}$ , entonces  $b_k$  siempre convergerá a éste<sup>1</sup>. Lo que es más, se podrá aproximar el autovalor dominante por medio del coeficiente de Rayleigh:

$$(2) \quad \lambda_{max} = \frac{b_k^t \mathbf{A} b_k}{b_k^t b_k}$$

MÉTODO DE LA DEFLACIÓN. El método de la deflación, por su parte, corresponde a la transformación de la matriz inicial  $\mathbf{A}$  por una matriz  $\mathbf{B}$  con autovalores equivalentes, salvo por el autovalor dominante que será anulado. Existen distintos métodos de deflación, entre ellos la deflación de Hotelling y la deflación de Wielandt [2].

En este trabajo utilizaremos la deflación de Hotelling por su sencillez, a cuestas de un mayor error numérico [2]. La misma consiste en aplicar el método de la potencia para sucesivas matrices que satisfagan la siguiente relación:

$$(3) \quad \mathbf{B}_{k+1} = \mathbf{B}_k - \lambda_k v_k v_k^t \quad \forall k \in \mathbf{N}_0$$

donde  $\mathbf{B}_0 = \mathbf{A}$ ,  $\lambda_k$  corresponde al  $k$ -ésimo autovalor en módulo máximo de  $\mathbf{A}$ , estimado por el método de la potencia, y  $v_k$  es su autovector asociado.

---

<sup>1</sup>De manera más formal, la secuencia definida por  $\{b_k\}_{k \in \mathbf{N}_0}$  es acotada para  $k$  y  $\lambda_{max}$  arbitrarios, pero la subsecuencia definida para  $k$  par es absolutamente convergente.

**1.2. Implementación.** Procedemos a detallar una posible implementación<sup>2</sup> para ambos métodos, restringiéndonos al caso de autovalores reales. Definimos:

*deflacion* :  $\text{matriz}_{n \times n} \mathbf{A} \times \text{nat } k \times \text{nat } q \times \text{real } t \longrightarrow \text{vector}_k \times \text{matriz}_{n \times k}$

*potencia* :  $\text{matriz}_{n \times n} \mathbf{A} \times \text{nat } q \times \text{real } t \longrightarrow \text{real} \times \text{vector}_n$

donde  $n$  es un natural,  $\mathbf{A}$  tiene al menos  $k$  autovalores reales dominantes en módulo,  $0 < k \leq n$ ,  $q$  es un número par<sup>3</sup> que representa el máximo de iteraciones a realizar y  $0 \leq t$  representa la tolerancia mínima a partir de la que se considera la convergencia de una solución.

---

```

1 proc deflacion(in A: matriz<n, n>, in k: Nat, in q: Nat, in t: Real) {
2
3     eigvals := vector<q>
4     eigvecs := matriz<n, k>
5
6     i := 0
7     while i < k {
8
9         l, v := potencia(A, q, t)
10        eigvals[i] := l
11        eigvecs.columna[i] := v
12
13        A := A - l * (v * v.t)
14        i := i + 1
15    }
16
17    return eigvals, eigvecs
18 }
```

---

ALGORITMO 1. Pseudocódigo para el método de la deflación.

El algoritmo (1.) retornará un vector con los primeros  $k$  autovalores en módulo máximos de  $\mathbf{A}$ , ordenados descendientemente, y una matriz cuyas columnas corresponden, respectivamente, a los autovectores normalizados asociados a estos autovalores.

Es interesante notar que la  $k$ -ésima matriz sobre la que se aplicará el método de la potencia — $\mathbf{B}_k$ — tendrá, por definición, un autovalor cero con multiplicidad algebráica mayor o igual a  $k$ . En el caso en que la matriz inicial sea simétrica,  $\mathbf{B}_k$  será simétrica<sup>4</sup> y, en consecuencia, diagonalizable. Se puede demostrar que la única matriz diagonalizable con multiplicidad algebráica  $\mu_a(0) = n$  es la matriz nula, por lo que el método de la deflación de Hotelling

---

<sup>2</sup>El código se puede encontrar en [./implementacion/src/](#).

<sup>3</sup>Esta restricción no es necesaria, pero permite mantener exacta la cantidad de iteraciones a realizar.

<sup>4</sup>Por suma de simétricas. Notar que  $vv^t$  siempre resulta en una matriz de este tipo.

tenderá hacia ella. Esto nos permite inferir que el error numérico será proporcional a  $k^5$ . Es decir, los autovalores más chicos de la matriz serán más susceptibles a errores.

Por su parte, el algoritmo (2.) retornará el autovalor de  $\mathbf{A}$  máximo en módulo y su autovector asociado:

---

```

1 proc potencia(in A: matriz<n, n>, in q: Nat, in t: Real) {
2
3     v := aleatorio(n)      // un vector aleatorio no nulo
4     v := v / norma(v, 2)   // ||v||_2 = 1
5
6     i := 0
7     while i < q / 2 {
8
9         y := A * (A * v)
10        y := y / norma(y, 2)
11        if norma(v - y, 2) < t { // tolerancia para la convergencia
12            break
13        }
14        v := y
15        i := i + 1
16    }
17
18    l := (v.t * A * v) / (v.t * v) // coeficiente de rayleigh
19
20    return l, v
21 }
```

---

ALGORITMO 2. Pseudocódigo para el método de la potencia *monte carlo*.

Observamos que el algoritmo (2.) trabaja con la subsecuencia par de  $\{b_k\}_{k \in \mathbb{N}_0}$ . Como mencionamos antes, esto garantiza la convergencia absoluta del método, dada una selección inicial de  $b_0 = v$  en nuestro pseudocódigo—no ortogonal al autovector asociado al autovalor en módulo dominante. Sin embargo, como esta selección depende de un proceso aleatorio, el algoritmo podrá resultar en una respuesta incorrecta. A este tipo de procesos se los denomina *monte carlo* [1].

Una variante posible, de tipo *las vegas*, se presenta en el algoritmo (3.). Hasta llegar a un resultado aceptable, determinado por  $\epsilon$ , o superar la cantidad de iteraciones permitida, definida por  $\alpha$ , el algoritmo volverá a intentar resolver el problema. De no alcanzar una respuesta aceptable retornará una señal de error.

De este modo, la probabilidad de fallar a causa de una selección inicial ortogonal al autovector al que se espera converger será inversamente proporcional a  $\alpha^6$ .

---

<sup>5</sup>En tanto existirá una correlación. Sin embargo, es esperable que otros factores entren en juego: la varianza de los autovalores, el número de condición de la matriz, la selección del vector aleatorio inicial, o la cantidad de iteraciones  $q$  a realizar, por ejemplo.

<sup>6</sup>Esto es, considerando que cada selección inicial es independiente.

Para reducir esta probabilidad desde el comienzo, se propone que se elija cada coordenada del vector de manera pseudo-aleatoria sobre un rango amplio<sup>7</sup>. Por ejemplo, la máxima representación de enteros con signo en 32 bits.

---

```

1 proc potencia'(in A: matriz<n, n>, in q: Nat, in t: Real,
2                 in alpha: Nat, in epsilon: Real) {
3
4     res := false
5
6     i := 0
7     do {
8
9         l, v := potencia(A, q, t)    // monte carlo
10        res := norma(A * v - v * l, 2) < epsilon
11        i := i + 1
12
13    } while not res and i < alpha
14
15    if !res {
16        return -1
17    }
18    return l, v
19 }
```

---

ALGORITMO 3. Pseudocódigo para el método de la potencia *las vegas*.

Es importante mencionar que una mala selección de  $\epsilon$  —en función de  $q$  y  $t$ — resultará en un algoritmo que siempre falla. Esto dependerá de la velocidad de convergencia del método para la matriz particular sobre la que se lo aplica.

**1.3. Evaluación cuantitativa.** Procedemos a realizar una evaluación de nuestra implementación en C++ acorde a los algoritmos propuestos<sup>8</sup>. Se optó por utilizar el método de la potencia *monte carlo*.

Medimos el error relativo  $\|\mathbf{A}\bar{\mathbf{V}} - \bar{\mathbf{V}}\bar{\Lambda}\|$  y absoluto  $\|\Lambda - \bar{\Lambda}\|$  para 300 instancias de matrices  $\mathbf{A} \in \mathbb{R}^{20 \times 20}$  generadas aleatoriamente, donde  $\bar{\mathbf{V}}$  y  $\bar{\Lambda}$  representan, respectivamente, las matrices aproximadas de autovectores y autovalores de  $\mathbf{A}$ .

METODOLOGÍA. Se calculó  $\bar{\Lambda}, \bar{\mathbf{V}} = deflacion(\mathbf{A}, 20, 2e^4, 1e^{-20})$  y se midió el error relativo y absoluto del resultado.

---

<sup>7</sup>Desde un punto de vista geométrico, dos vectores son ortogonales sólo si son perpendiculares. De manera intuitiva, podemos ver que a medida que la dirección inicial posible de un vector tiende al infinito, la probabilidad que forme un ángulo de 90 grados con otro tiende a cero.

<sup>8</sup>El script asociado se puede encontrar en `./experimentos/error-potencia.py`, las tablas resultantes en `./experimentos/resultados/error-potencia`.

Cada caso se generó a través de uno de los siguientes tres procedimientos<sup>9</sup>:

- 1) *Matrices Diagonales*: Se generaron cien matrices diagonales  $\mathbf{D}$  con cien autovalores en el rango  $[-1e^3, 0] \cup (0, 1e^3]$ , con paridad diferente acorde al signo<sup>10</sup>. Los mismos se generaron con el rng *PCG64* de numpy.
- 2) *Matrices Diagonalizables*: Se generaron cien matrices diagonalizables  $\mathbf{A} := \mathbf{Q}\mathbf{D}\mathbf{Q}^t$  donde cada matriz  $\mathbf{D}$  se generó a partir de la metodología (1.) y  $\mathbf{Q} := \mathbf{I} - 2uu^t$  se generó a partir de un vector aleatorio  $u$  —con el algoritmo *random.rand()* de numpy— tal que  $\|u\|_2 = 1$ .
- 3) *Matrices Simétricas Definidas Positivas*: Se generaron cien matrices simétricas definidas positivas de enteros  $\mathbf{A} := \mathbf{B}\mathbf{B}^t$  donde  $\mathbf{B}$  es una matriz aleatoria generada con el algoritmo *random.randint()* de numpy para el rango  $[-1e^3, 0] \cup (0, 1e^3]$ .

**OBSERVACIONES.** Consideramos que la varianza de los autovalores, el número de condición de las matrices, y su tamaño, son variables que afectan de manera significativa en el error del procedimiento.

El proceso mencionado para la generación de matrices aleatorias fue pensado sobre generadores de números aleatorios para tratar de minimizar cualquier tendencia que pueda provenir de la utilización de distribuciones particulares. De esta manera se espera que la varianza de los autovalores y el número de condición de las matrices también sean aleatorios, tal que los resultados brinden un panorama amplio del dominio de aplicación del método propuesto.

Además, se controló el tamaño de  $n$ , la cantidad de iteraciones  $q$  y la tolerancia  $t$ . Un estudio más exhaustivo debería también evaluar el comportamiento del algoritmo en función de estos parámetros.

**RESULTADOS.** La Figura (1.) resume los resultados para el error relativo.

test	$L_1$	$L_\infty$
mediciones	300	300
error relativo promedio	$2.20e^{-3}$	$3.69e^{-4}$
desviación estándar	$3.36e^{-2}$	$5.83e^{-3}$
mínimo	0.0	0.0
25%	0.0	0.0
50%	$2.00e^{-11}$	$1.01e^{-11}$
75%	$4.13e^{-6}$	$7.97e^{-7}$
máximo	$5.81e^{-1}$	$1.01e^{-1}$

FIGURA 1. Datos de resumen para el error relativo  $\|\mathbf{A}\bar{\mathbf{V}} - \bar{\mathbf{V}}\bar{\Lambda}\|$ .

---

<sup>9</sup>Se utilizó un valor semilla para facilitar la reproductibilidad.

<sup>10</sup>De esta manera se garantiza la dominancia estricta en módulo de los autovalores asociados a la matriz.

La Figura (2.), por su parte, resume los resultados para el error absoluto.

test	$L_1$	$L_\infty$
mediciones	300	300
error relativo promedio	$3.43e^{-7}$	$2.05e^{-7}$
desviación estándar	$5.05e^{-6}$	$3.31e^{-6}$
mínimo	0.0	0.0
25%	0.0	0.0
50%	$2.20e^{-12}$	$4.55e^{-13}$
75%	$1.31e^{-7}$	$2.98e^{-8}$
máximo	$8.75e^{-5}$	$5.73e^{-5}$

FIGURA 2. Datos de resumen para el error absoluto  $\|\Lambda - \bar{\Lambda}\|$ .

Vemos que el método funcionó, para el 75% de las matrices evaluadas, con un error relativo y absoluto menor a  $1e^{-4}$ . Sin embargo, notamos que existen casos anómalos para los que el error puede ser mayor.

Por la disparidad entre el error relativo y absoluto, podemos inferir —a modo de futura hipótesis a investigar— que el método produce más error en los autovectores resultantes que en los autovalores.

## 2. ANÁLISIS: CLUB DE KARATE

**2.1. Contexto.** La red del *Club de Karate* fue parte de una investigación antropológica [6], realizada por Wayne W. Zachary, que estudió las relaciones ‘políticas’ entre los miembros de un club universitario. La misma se realizó durante el desarrollo de un conflicto que terminó por dividir al grupo.

La red buscó modelar el flujo de información entre sus integrantes por medio de la tripla (**V**, **E**, **C**), donde **V** denota el conjunto de individuos, **E** refiere a un grafo no dirigido cuyos ejes representan los vínculos entre los miembros del club, y **C** define la fuerza de estas relaciones —lo que se podría pensar como ponderaciones sobre los ejes de **E**—.

La investigación tuvo como enfoque central demostrar la capacidad del modelo para predecir la división del grupo por medio del algoritmo de *labeling* de ‘flujo máximo - corte mínimo’ de Ford y Fulkerson [3].

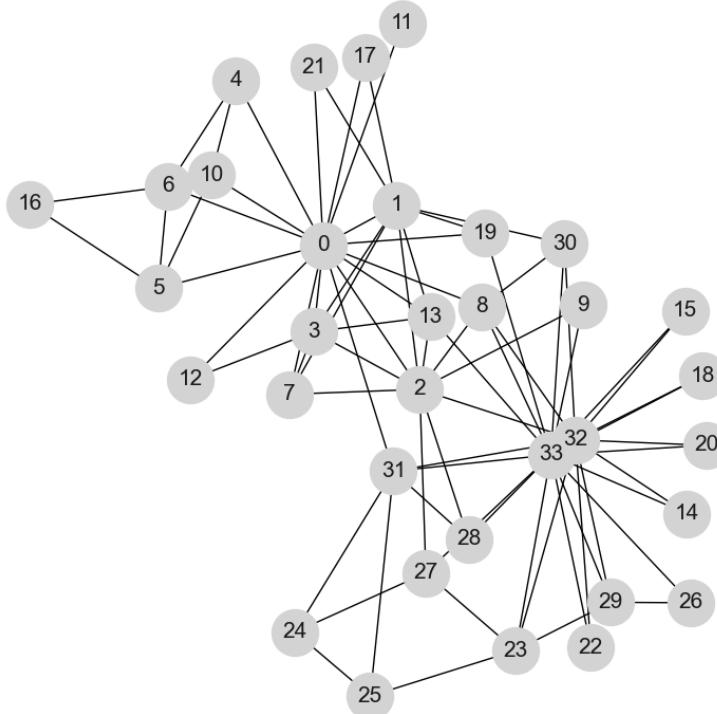


FIGURA 3. La red del *Club de Karate*. Cada nodo representa un individuo, cada eje la existencia de un vínculo por fuera del club.

En este análisis<sup>11</sup> utilizaremos la representación matricial de **E** para evaluar la importancia de los distintos miembros en la red, y la matriz laplaciana asociada para evaluar el uso de autovectores como predictores de la división del grupo.

<sup>11</sup>El script asociado se puede encontrar en `./experimentos/club_de_karate.py`, los archivos con los resultados en `./experimentos/resultados/club-karate/`.

**2.2. Centralidad de Autovector.** La centralidad de autovector es una medida que se utiliza en el análisis de redes para evaluar la ‘importancia’ de los nodos que componen una red, relativa a la importancia de sus conexiones. Dada una matriz de conectividad  $\mathbf{W} \in \mathbb{R}^{n \times n}$ , se define:

$$(4) \quad \lambda x = \mathbf{W}x$$

donde  $\lambda$  es el autovalor en módulo máximo de  $\mathbf{W}$  y la coordenada  $x_i$  —del autovector  $x$  asociado a  $\lambda$ — denota la centralidad del nodo  $i$ ,  $\forall i : 1 \dots n$ .

Intuitivamente, se puede pensar que la importancia de cada nodo es proporcional a la suma de las importancias de sus vecinos. Se puede demostrar [5] que, dado las características de esta matriz, el autovector asociado tendrá el mismo signo en todas sus coordenadas.

En tanto la red de *Club de Karate*, podemos ver que la aplicación del método de la potencia sobre la matriz de conectividad asociada al grafo  $\mathbf{E}$  resulta en el siguiente autovector  $x^{12}$ :

0	0.355491	17	0.092400
1	0.265960	18	0.101403
2	0.317193	19	0.147913
3	0.211180	20	0.101403
4	0.075969	21	0.092400
5	0.079483	22	0.101403
6	0.079483	23	0.150119
7	0.170960	24	0.057052
8	0.227404	25	0.059206
9	0.102674	26	0.075579
10	0.075969	27	0.133477
11	0.052856	28	0.131078
12	0.084255	29	0.134961
13	0.226473	30	0.174758
14	0.101403	31	0.191034
15	0.101403	32	0.308644
16	0.023636	33	0.373363

FIGURA 4. Centralidad de autovector para la red del *Club de Karate*. La columna izquierda denota el nodo, la derecha su ‘importancia’.

Vemos que el nodo ‘0’ y el nodo ‘33’ son los más centrales. Esto no es casualidad, la red del *Club de Karate* está armada para tener a las dos figuras principales del conflicto en cada extremo —el instructor de karate y el presidente del club, respectivamente— para satisfacer la especificación del algoritmo de labeling que utiliza.

---

<sup>12</sup>Notamos que el error relativo del resultado fue de  $\approx 7.81e^{-14}$  en norma  $L_1$ .

**2.3. Autovectores de la matriz laplaciana.** La matriz laplaciana  $\mathbf{L} = \mathbf{D} - \mathbf{W}$  —donde  $\mathbf{D}$  es una matriz diagonal con elementos  $d_{ii} = \sum_j w_{ij}$  y  $\mathbf{W}$  es una matriz de conectividad— sirve para medir distintas propiedades en una red.

En particular, el mínimo autovalor en módulo no nulo —llamado de conectividad algebráica, o valor de Fiedler— permite establecer un criterio sobre el que particionar la red en dos. El autovector asociado a este autovalor designará la pertenencia de un nodo a una u otra partición acorde a su signo.

Procedemos a analizar qué autovector de la matriz laplaciana asociada a  $\mathbf{E}$  permite predecir mejor la división que ocurrió en el *Club de Karate*. Para ello, calculamos los autovectores de la matriz con el método de la potencia con deflación<sup>13</sup> y medimos el valor absoluto de la correlación entre cada autovector y un vector que indica la división verdadera que ocurrió en el grupo.

18.136696	0.045692	3.013963	0.009556
17.055171	0.011443	2.749157	0.161363
13.306122	0.078413	2.487092	0.117379
10.921068	0.058283	2.0	0.0
9.777241	0.085078	2.0	0.0
6.996197	0.010429	2.0	0.0
6.515545	0.079173	2.0	0.0
6.331592	0.014177	2.0	0.0
5.618034	0.0	1.955050	0.011172
5.378595	0.244844	1.826055	0.068783
4.580793	0.079869	1.761899	0.027133
4.480008	0.044972	1.599283	0.05576
4.275877	0.010777	1.259404	0.000408
3.472187	0.000321	1.125011	0.333307
3.381966	0.0	0.909248	0.265918
3.376154	0.065159	0.468525	0.814727
3.242067	0.086678	0.0	nan <sup>14</sup>

FIGURA 5. Correlación entre los autovectores asociados a los autovalores de la red del *Club de Karate* y un vector que indica la división verdadera que ocurrió en el grupo. La columna izquierda denota el autovalor, la derecha la correlación.

Vemos que el valor de conectividad algebráica —el autovalor mínimo en módulo no nulo— tiene asociado el mejor autovector para realizar el corte. En la figura (6.) se puede observar que sólo dos nodos obtuvieron una clasificación incorrecta: el ‘2’ y el ‘8’, lo que presenta un desempeño casi igual al logrado en la investigación original<sup>15</sup>.

<sup>13</sup>Notamos que el error relativo de los resultados tuvo una cota superior de  $\approx 1.03e^{-13}$  en norma  $L_1$ .

<sup>14</sup>El método de la potencia no está definido si el autovalor máximo en módulo es nulo. Este resultado da cuenta de ello. Como nos interesan solo los autovalores diferentes a ‘0’, optamos por descartarlo.

<sup>15</sup>En este sentido, solo el nodo ‘2’ recibió una clasificación incorrecta. Sobre el ‘8’, Zachary considera [6] que hubo un interés ‘egoísta’ que primó sobre la ‘afiliación’ política (es decir, las relaciones) del nodo.

0	-0.112137	0	0
1	-0.041288	0	0
2	0.023219	1	0
3	-0.055500	0	0
4	-0.284605	0	0
5	-0.323727	0	0
6	-0.323727	0	0
7	-0.052586	0	0
8	0.051601	1	0
9	0.092801	1	1
10	-0.284605	0	0
11	-0.210993	0	0
12	-0.109461	0	0
13	-0.014742	0	0
14	0.162751	1	1
15	0.162751	1	1
16	-0.422765	0	0
17	-0.100181	0	0
18	0.162751	1	1
19	-0.013637	0	0
20	0.162751	1	1
21	-0.100181	0	0
22	0.162751	1	1
23	0.155695	1	1
24	0.153026	1	1
25	0.160963	1	1
26	0.187110	1	1
27	0.127664	1	1
28	0.095152	1	1
29	0.167650	1	1
30	0.073500	1	1
31	0.098753	1	1
32	0.130345	1	1
33	0.118903	1	1

FIGURA 6. Autovector de Conectividad algebráica, la primer columna representa los nodos de la red, la segunda el autovector, la tercera la clasificación de cada nodo acorde a su signo y la cuarta la división verdadera que ocurrió.

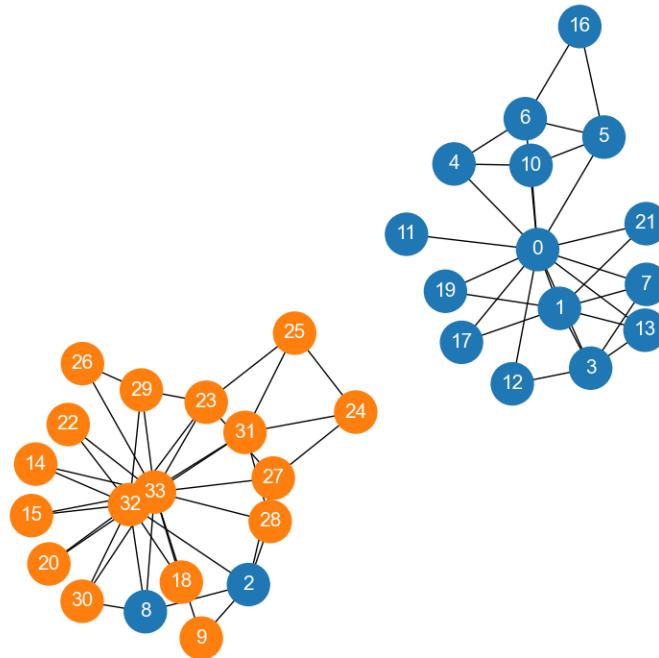
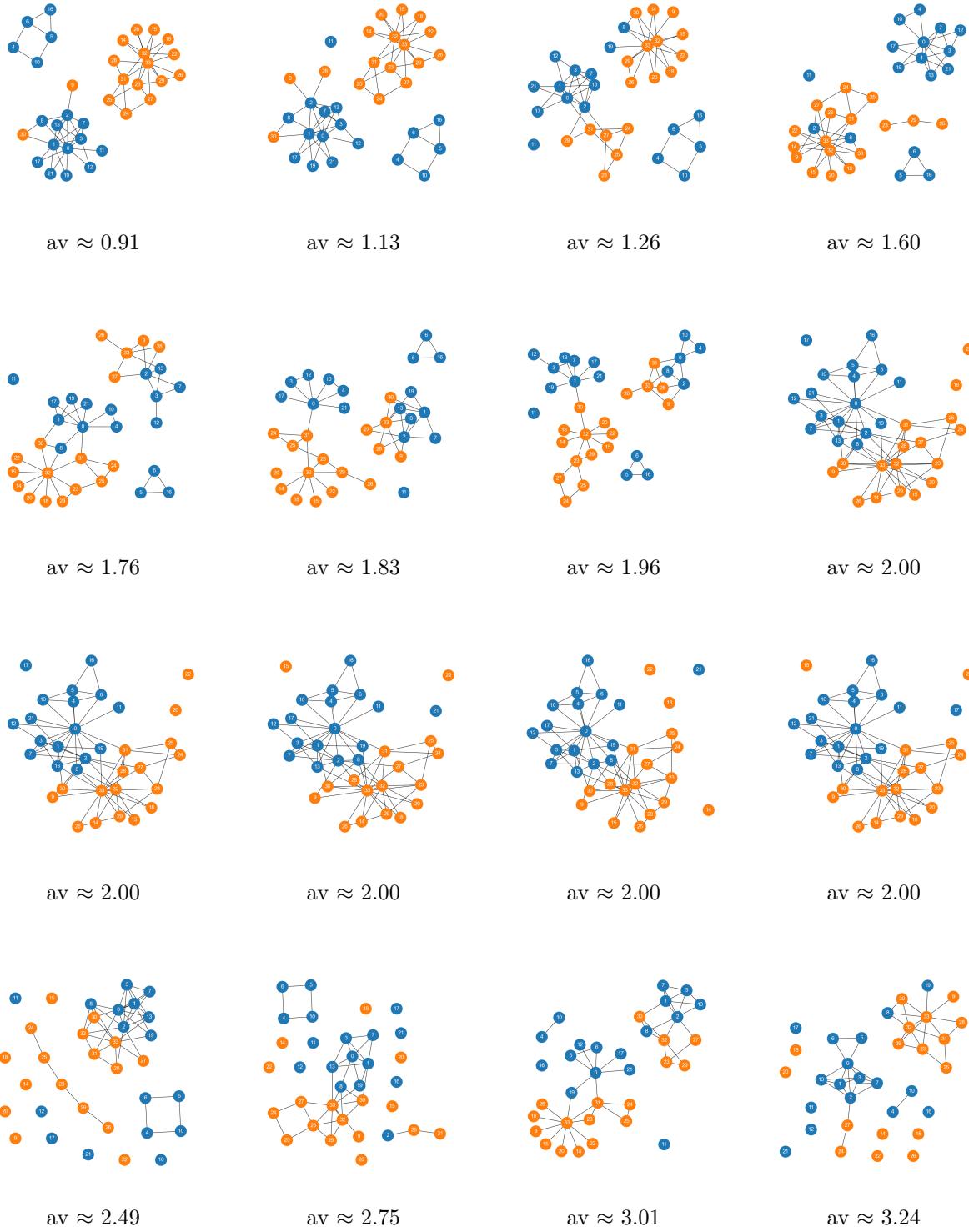


FIGURA 7. La división estimada de la red del *Club de Karate*. Los colores representan la división verdadera de la red. Azul: '0', Naranja: '1'.

Procedemos a detallar —a modo de comparación visual— los cortes realizados por los demás autovectores. Notamos que el proceso de corte sólo elimina los ejes existentes entre pares cuya clasificación difiere. Por ello, la cantidad de componentes conexos que puede generar un corte no es necesariamente dos.



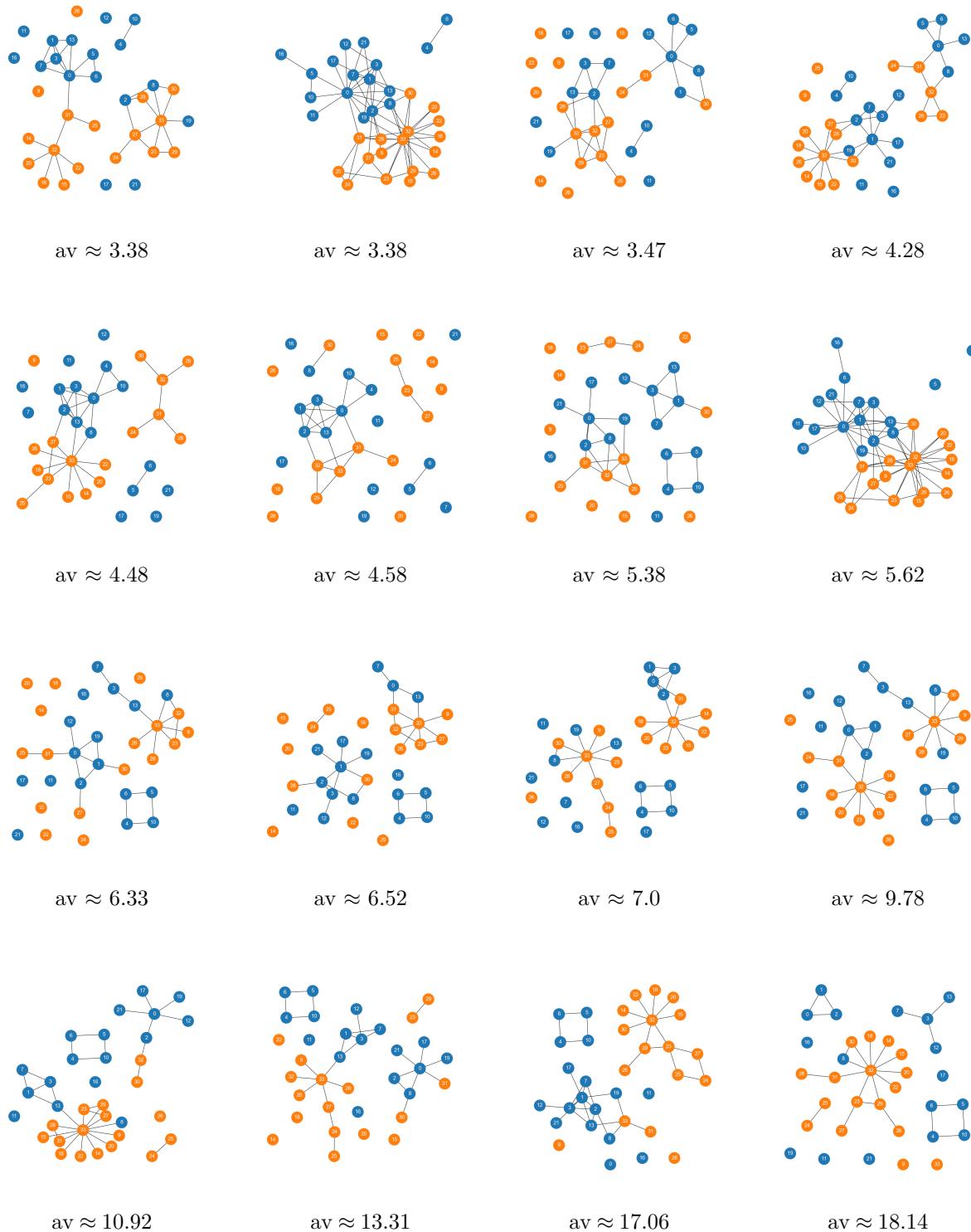


FIGURA 7. Todos los cortes posibles —salvo el de Fiedler y el nulo—, correspondientes a los autovectores asociados a la matriz laplaciana de la red del *Club de Karate*. Cada uno se designa por medio de su autovalor.

### 3. ANÁLISIS: RED ‘EGO’

**3.1. Contexto.** En el análisis de redes sociales, una *ego-network* [4] es una red compuesta por las amistades que existen entre los amigos de un individuo, el ‘ego’. Estas redes son centrales en aplicaciones como Facebook, Google+ o Twitter.

En particular, dada una red de estas características, resulta de interés poder identificar los círculos sociales —conjuntos, disjuntos y anidados— a los que pertenece un usuario. Leskovec [4] propone un método de aprendizaje no supervisado para lograr inferirlos, que se nutre de la siguiente información: un grafo **E** (la red) —donde se espera que exista una correlación fuerte entre un círculo y la densidad de conexiones entre los nodos que lo componen— y un conjunto de atributos **C** para cada nodo —donde se espera que exista una correlación entre un círculo y la similaridad de los atributos de los nodos que lo componen—.

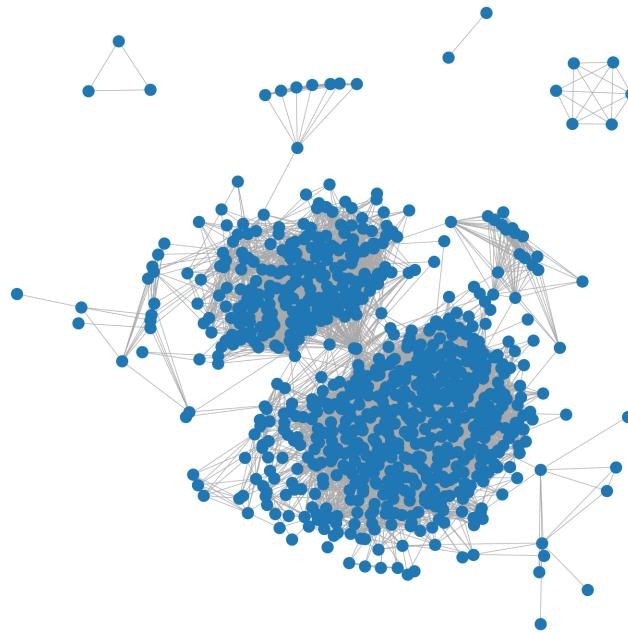


FIGURA 8. Una *ego-network* de Facebook compuesta por 786 nodos. Cada vértice representa un individuo, cada arista la existencia de una amistad.

En este análisis<sup>16</sup> estimaremos la red ‘ego’ **E** de la figura (8.), por medio de la construcción de una matriz de similaridad que utilice el conjunto de atributos **C** asociados a **E**<sup>17</sup>. También, buscaremos reducir su dimensionalidad por medio del análisis de componentes principales.

<sup>16</sup>El script asociado se puede encontrar en `./experimentos/ego_facebook.py`, los archivos con los resultados en `./experimentos/resultados/ego-facebook/`.

<sup>17</sup>Por cuestiones de privacidad, los datos están despojados de cualquier referencia a sus cualidades: los nodos y atributos se distinguen por medio de números; los atributos, además, por una parametrización binaria que designa si el nodo porta o no cierta calidad anónima.

**3.2. Matriz de similaridad.** Queremos generar una aproximación de nuestra red ‘ego’ original,  $\mathbf{E}$ , a partir de los atributos de los usuarios que la componen, definidos en  $\mathbf{C}$ . Es decir, buscamos un método que nos permita comparar los atributos de dos nodos diferentes y, bajo algún criterio propuesto, decidir conectarlos o no con el fin de replicar la red verdadera.

Intuitivamente, una forma de decidir si dos usuarios deberían estar conectados, o no, es contar la cantidad de atributos que comparten. Se podría pensar que si coinciden en muchos, entonces existe una mayor probabilidad que pertenezcan al mismo círculo. Otra forma podría ser medir la distancia, a partir de alguna norma, entre sus vectores de atributos.

De manera general, podemos pensar en las *matrices de similaridad*. Dado un conjunto de datos  $X$ , se aplica una función a cada par de elementos  $x_i, x_j$  tal que:

$$(5) \quad s_{ij} = f(x_i, x_j)$$

En nuestro caso, trabajaremos con una tabla de atributos<sup>18</sup> cuya primer columna contiene las etiquetas (*tags*) de cada nodo en la red, y el resto de las columnas forman  $\mathbf{C}$  —donde la fila <sub>$i$</sub> ( $\mathbf{C}$ ) representa los atributos del usuario definido por la etiqueta  $i$ —, y definiremos la matriz de similaridad de la siguiente forma:

$$(6) \quad \mathbf{S} = \mathbf{C} \mathbf{C}^t$$

donde  $s_{ij}$  expresa la similaridad entre el nodo  $i$  y el  $j$  por medio del producto interno de sus respectivos vectores de atributos.

Con esta información propondremos diferentes umbrales  $u$  en el rango  $[min(\mathbf{S}), max(\mathbf{S})]$  que servirán como criterio para aproximar  $\mathbf{E}$ . Si  $s_{ij} > u$ , entonces los nodos  $i$  y  $j$  estarán conectados.

**RESULTADOS.** Vemos que cada elemento  $s_{ij}$  es entero, por producto punto de enteros, y  $[min(\mathbf{S}), max(\mathbf{S})] = [0, 22]$ . A partir de estas observaciones aproximamos  $\mathbf{E}$  por medio de 22 umbrales distintos:  $[0, 1, \dots, 22]$ . La figura (9.) muestra una representación de los grafos obtenidos más significativos.

Observamos que cuanto mayor sea el umbral, menos conexiones tendremos en nuestra aproximación. En particular, con  $u = 0$  se obtiene un grafo donde casi todos los usuarios están conectados entre sí, y con  $u > 12$  obtenemos un grafo sin ninguna arista. Nos enfocaremos entonces en los grafos asociados a  $u \in [0, 12]$ , ya que estos son los que revelan la mayor cantidad de información.

---

<sup>18</sup>La misma se puede encontrar en `./catedra/ego-facebook.feat`. Dado que la red original y esta tabla no comparten todos sus nodos, se realizó un proceso de filtrado y limpieza. Las versiones corregidas se encuentran en `./experimentos/resultados/ego-facebook/in/`.

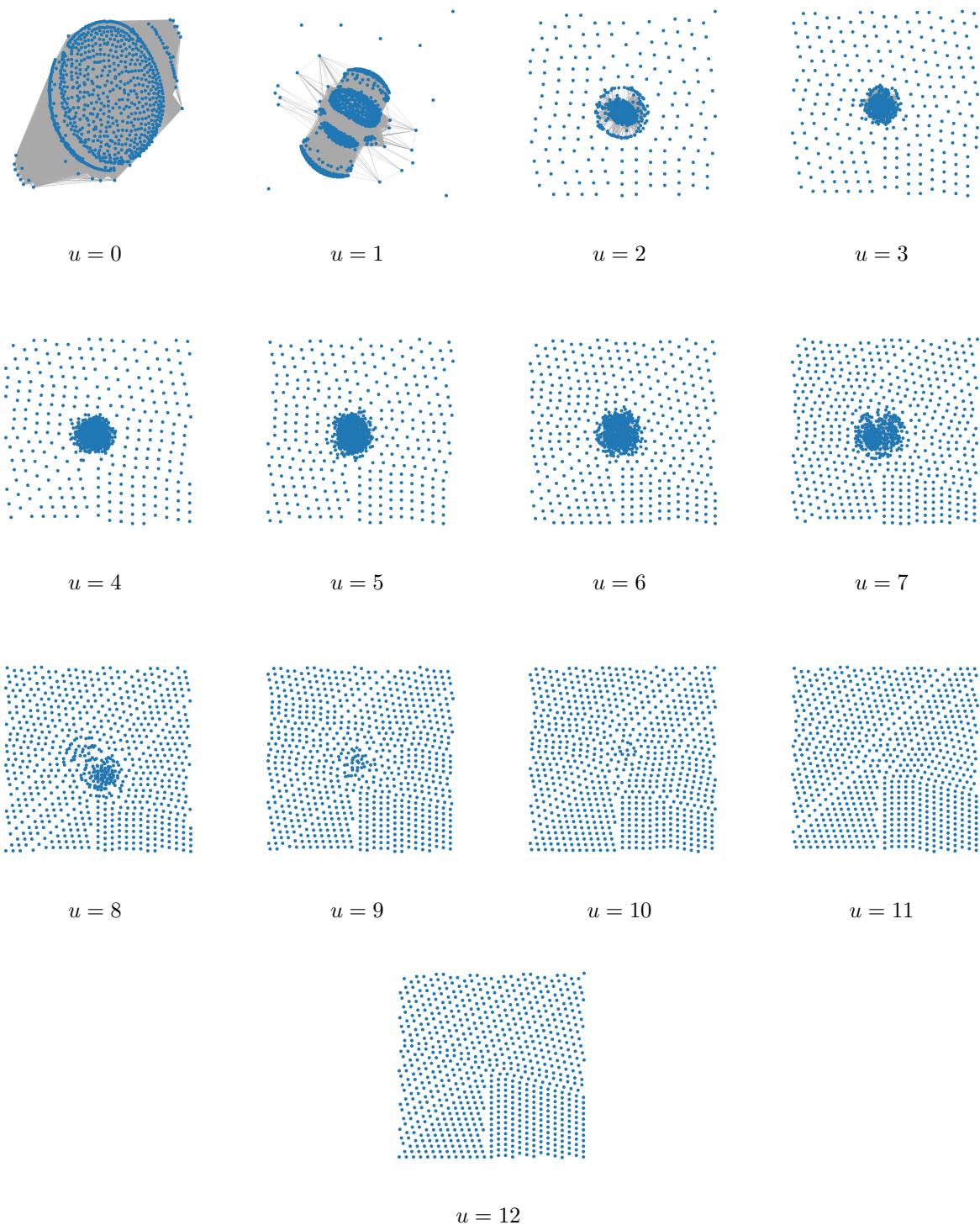


FIGURA 9. Todos los grafos correspondientes a los diferentes umbrales, tomados a partir de las matrices de conectividad  $\mathbf{C} \mathbf{C}^t > u$ . Los números debajo de cada imagen indican el umbral utilizado.

**3.3. Comparación con la red original.** Nos encontramos con diferentes grafos construidos en base a los atributos en **C**. Buscamos ahora comparar nuestras aproximaciones con la red original.

Un primer acercamiento a este problema puede ser calcular la cantidad de *posiciones coincidentes* entre las matrices de adyacencia y dividir por los elementos totales. La lógica por detrás del método es que nuestros grafos serán más similares entre sí por cada conexión y ‘no conexión’ acertada.

Otros dos métodos de comparación posibles incluyen *la correlación de las matrices de adyacencia estiradas* —donde se computa la correlación entre los vectores estirados de las matrices—, y *la correlación de las listas de autovalores* —donde se computa la correlación entre los vectores de autovalores de las matrices, ordenados—.

Por su parte, la correlación es una covarianza normalizada con valores en  $(-1, 1)$ , donde números mayores indican una mayor similitud. Se describe por la siguiente fórmula:

$$(7) \quad \text{Corr}(x, y) = \frac{(x - \mu_x) \cdot (y - \mu_y)}{\sqrt{(x - \mu_x)^2 \cdot (y - \mu_y)^2}}$$

siendo  $\mu$  el valor medio de los vectores.

aproximación	conexiones	coincidentes	adyacencia estirada	autovalores
0	298,799	7.61%	1.25%	47.99%
1	213,902	32.95%	2.32%	60.86%
2	127,382	58.57%	2.94%	62.98%
3	86,847	70.99%	5.86%	75.48%
4	43,091	84.16%	9.61%	88.29%
5	16,850	91.54%	10.96%	94.28%
6	5,727	94.32%	9.75%	95.06%
7	1,517	95.21%	6.90%	92.64%
8	405	95.40%	4.15%	85.62%
9	108	95.45%	2.34%	77.52%
10	36	95.46%	1.35%	63.31%
11	8	95.46%	0.50%	55.00%
12	2	95.46%	0.055%	46.19%
13	0	95.46%	0.0%	0.0%

FIGURA 10. Similitud elemento a elemento de las matrices de adyacencia, la primer columna representa el umbral tomado para la aproximación y la segunda la cantidad de aristas del grafo (la red ‘ego’ original cuenta con 14.024). Las columnas siguientes miden el porcentaje de similitud respecto a **E**, para los distintos métodos propuestos.

**RESULTADOS.** Como se puede observar en la figura (10.), los grafos con muy pocas conexiones (por ejemplo, para  $u > 10$ ), tienen los valores más altos de similitud por *pocisiones*

*coincidentes*, mientras que otros con cantidad de aristas similares a la red original (como  $u = 5$  o  $u = 6$ , que cuentan con 16.850 y 5.727 conexiones respectivamente), parecerían ser peores aproximaciones. Esto se da por la naturaleza rala de nuestras matrices de adyacencia. La de la red ‘ego’  $\mathbf{E}$  es de  $786 \times 786$ , con 617.796 elementos, y tan solo 28.048 (4.54%) no nulos. Es decir, es rala en un 95.46%. Es por esto que comparar elemento a elemento no prueba ser un método muy descriptivo de qué tan buena es una aproximación, ya que cualquiera con pocos elementos coincidirá en un  $\sim 95\%$ .

En la figura (11.) se puede apreciar que se obtienen resultados considerablemente diferentes a los de nuestro primer método de comparación si se emplean los otros. En un primer lugar, siguiéndonos de la correlación entre los vectores de *adyacencia estirada*, parecería que las mejores aproximaciones son, razonablemente, las que tienen una cantidad de aristas similar a la red original (ver 9.), con los umbrales  $u \in [4, 7]$ , mientras que los grafos vacíos con umbrales más altos pasan a tener correlación 0. De la misma forma, analizando con las *listas de autovalores* se obtienen resultados similares, donde la mayor correlación se halla con los umbrales de valores medios y decae avanzando para los extremos.

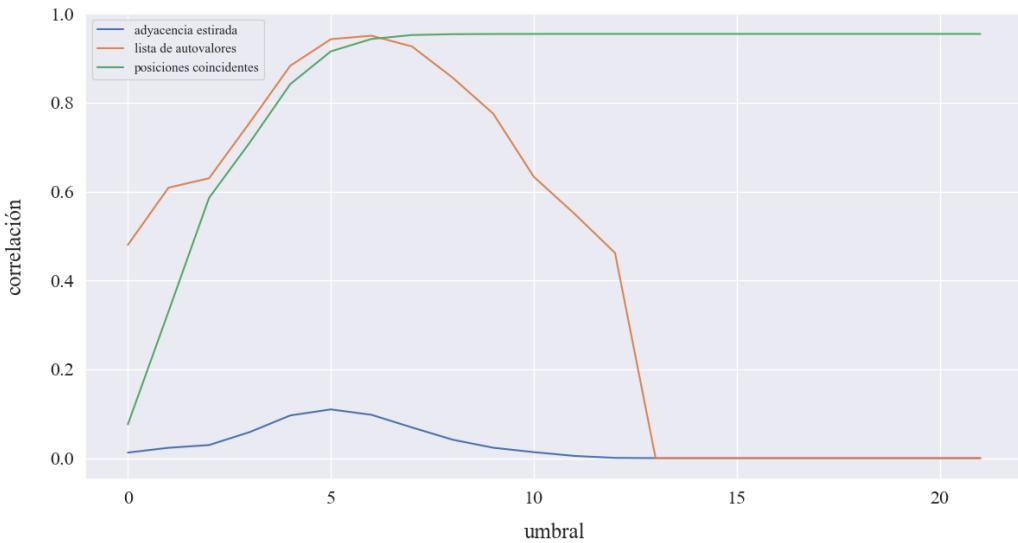


FIGURA 11. Similitud entre las aproximaciones y el grafo  $\mathbf{E}$  para *posiciones coincidentes*, *adyacencia estirada* y *listas de autovalores*, en función del umbral  $u$ .

¿Cuál es, entonces, el valor para  $u$  que genera la mejor aproximación? Observando la figura (11.), se puede deducir que los mejores umbrales son  $u = 5$  y  $u = 6$ . Los grafos formados bajo estas condiciones parecerían tener las mejores correlaciones con la red ‘ego’ verdadera, sobretodo comparándolos con los de umbrales muy bajos ( $u < 2$ ) o muy altos ( $u > 10$ ). Esto tiene sentido dado que mantienen una cantidad de conexiones similares al grafo original, característica observable en la figura (9.), y los demás umbrales parecerían tener muy pocas aristas (hasta ninguna) o demasiadas.

Sin embargo, notamos que los resultados son considerablemente distintos a la red original. El método de aproximación por matriz de similaridad propuesto no fue capaz, en nuestra opinión, de capturar la estructura de la red original.

**3.4. PCA.** Ahora bien, el conjunto describe 319 atributos distintos. En cada caso su significado puede variar notablemente. Su impacto sobre el sistema de conexiones puede diferir de tal manera que el subconjunto de atributos en consideración sea relevante para nuestra exploración del mismo, y tal que parte de los datos sea poco pertinente para dicho estudio. El Análisis de Componentes Principales (PCA) es un método que podemos aprovechar para reducir la dimensionalidad de los datos. Reduciendo la cantidad bajo consideración podemos obtener un conjunto de atributos que sea más relevante para la exploración del sistema de conexiones. PCA proyecta los datos en un espacio de menor dimensionalidad, donde los atributos son linealmente independientes, formando una base de vectores ordenados según su varianza de mayor a menor. Es decir, PCA organiza la información según su relevancia. Esto podría eventualmente ayudarnos a reducir la redundancia de los datos, y por ende, la cantidad de atributos que debemos considerar. Así se espera intercambiar un poco de precisión, deshaciéndonos de información redundante, para ganar simplicidad y eficiencia computacional.

Para lograr esto, se busca hacer un cambio de base a la matriz de datos  $\mathbf{C}$ . Y para eso vamos a diagonalizar la matriz de covarianza  $\mathbf{M}_x$ . Esta contiene en su diagonal los valores de varianza de cada variable, y la covarianza (definida en 7) en el resto de sus posiciones. La obtenemos de la siguiente manera:

$$(8) \quad \mathbf{M}_x = \frac{\mathbf{C}^t \mathbf{C}}{n - 1}$$

Donde  $n$  es la cantidad de nodos en el grafo.

Buscamos diagonalizar esta matriz tal que, para  $\mathbf{M}_x = \mathbf{V}\mathbf{D}\mathbf{V}^t$ ,  $\mathbf{V}$  contiene los autovectores de  $\mathbf{M}_x$  en sus columnas. Permutamos las columnas de  $\mathbf{V}$  para que estén ordenadas de mayor a menor según los valores de la diagonal de  $\mathbf{D}$ , los autovalores de  $\mathbf{M}_x$ , y obtenemos  $\mathbf{V}^*$ . La matriz resultante con base cambiada es  $\mathbf{C}^* = \mathbf{C}\mathbf{V}^*$ .

Podemos limitar el subconjunto de componentes principales a considerar. Definimos una variable  $p$  como el porcentaje de datos que deseamos preservar en nuestro conjunto de datos. El tamaño de la matriz  $\mathbf{V}^*$  va a depender de este valor. Así procedemos a trabajar con el valor  $p$  y el umbral  $u$  para estudiar nuevamente la aproximación de  $\mathbf{E}$ . Usando  $\mathbf{C}^*$  construimos nuevamente la matriz de similaridad e iteramos sobre el rango de umbrales.

La totalidad de los resultados obtenidos se adjuntan en el apendice (ver figuras 14 y 15). En la figura 12, ilustramos nuevamente los distintos métodos de comparación para cada iteración de  $p$ . Para los valores de  $p$  más pequeños, el gráfico difiere notablemente de los obtenidos en la sección anterior. Pero vemos que, como es de esperarse, los resultados tienden a converger al mismo comportamiento observado previamente.

Lo más interesante es que, con menos atributos, el valor de correlación máximo difiere solo ligeramente. Y este llega a su tope para un umbral  $u$  relativamente menor. La tabla 13

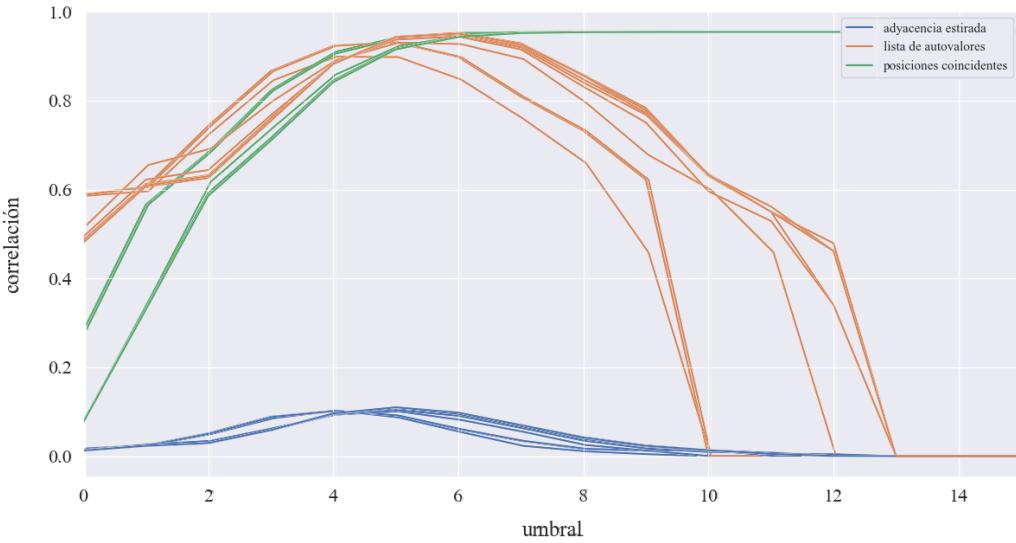


FIGURA 12. Similitud entre las aproximaciones y el grafo  $\mathbf{E}$  para *posiciones coincidentes*, *adyacencia estirada* y *listas de autovalores*, en función del umbral  $u$ , para  $p \in \{0.4, 0.5, 0.6, 0.75, 0.8, 0.85, 0.9, 0.95, 0.99\}$ .

muestra exactamente para qué valor de  $u$  se alcanza la máxima correlación, para cada valor de  $p$ , para los métodos de *adyacencia estirada* y *listas de autovalores*.

p	adyacencia estirada		lista de autovalores	
	u	máximo	u	máximo
0.4	4	10.29%	4	89.88%
0.5	4	10.18%	5	93.12%
0.6	5	10.8%	5	93.05%
0.75	5	10.4%	6	94.42%
0.8	5	10.49%	6	94.7%
0.85	5	10.81%	6	94.94%
0.9	5	10.96%	6	95.2%
0.95	5	10.99%	6	95.26%
0.99	5	10.99%	6	95.06%

FIGURA 13. Umbral con correlación máxima para valores de  $p$  en los métodos de *adyacencia estirada* y *listas de autovalores*.

Efectivamente, para menor valores de  $p$  empeora nuestra capacidad de aproximar  $\mathbf{E}$ , pero solo levemente. Notar que habiendo reducido la cantidad de atributos en un 60%, perdemos no más que 0.7% y 5% en la adyacencia estirada y la lista de autovalores, respectivamente. Es decir, de necesitar el uso del PCA para reducir la dimensionalidad de los datos, los resultados obtenidos sufren meramente una leve pérdida de calidad. También, constatamos que el rango de umbrales en el que se alcanza la máxima correlación converge a los mismos valores que en la sección anterior. Es decir, para  $u \in \{5, 6\}$ .

#### 4. CONCLUSIONES

## 5. APÉNDICE

p	atributos	u	conexiones	coincidentes	adyacencia estirada	autovalores
0.5	12	0	229,602	28.19%	1.73%	58.84%
0.5	12	1	133,807	56.57%	2.37%	60.61%
0.5	12	2	95,224	68.42%	5.12%	74.32%
0.5	12	3	49,381	82.28%	8.86%	86.65%
0.5	12	4	20,125	90.58%	10.18%	92.33%
0.5	12	5	6,716	94.04%	9.15%	93.12%
0.5	12	6	1,727	95.14%	6.08%	89.81%
0.5	12	7	393	95.4%	3.45%	80.9%
0.5	12	8	95	95.44%	1.66%	73.24%
0.5	12	9	24	95.46%	1.22%	62.23%
0.75	45	0	298,220	7.79%	1.34%	49.38%
0.75	45	1	212,612	33.34%	2.36%	62.21%
0.75	45	2	125,175	59.22%	2.97%	64.39%
0.75	45	3	84,766	71.6%	5.85%	76.71%
0.75	45	4	41,516	84.57%	9.36%	88.34%
0.75	45	5	15,922	91.73%	10.4%	93.74%
0.75	45	6	5,274	94.4%	9.07%	94.42%
0.75	45	7	1,342	95.24%	6.36%	91.53%
0.75	45	8	333	95.41%	3.5%	83.16%
0.75	45	9	89	95.45%	1.74%	75.06%
0.75	45	10	28	95.46%	0.94%	59.58%
0.75	45	11	8	95.46%	0.81%	52.95%
0.75	45	12	1	95.46%	0.04%	34.07%
0.8	60	0	298,611	7.66%	1.26%	48.59%
0.8	60	1	213,405	33.09%	2.31%	61.25%
0.8	60	2	126,336	58.87%	2.89%	63.2%
0.8	60	3	85,825	71.28%	5.82%	76.06%
0.8	60	4	42,266	84.36%	9.35%	88.29%
0.8	60	5	16,324	91.63%	10.49%	93.97%
0.8	60	6	5,488	94.35%	9.22%	94.7%
0.8	60	7	1,423	95.23%	6.53%	91.99%
0.8	60	8	364	95.4%	3.65%	84.08%
0.8	60	9	101	95.45%	2.01%	76.82%
0.8	60	10	34	95.46%	1.25%	63.04%
0.8	60	11	8	95.46%	0.5%	55.0%
0.8	60	12	1	95.46%	0.04%	34.07%
0.85	81	0	298,815	7.6%	1.22%	48.0%
0.85	81	1	213,850	32.96%	2.28%	60.65%
0.85	81	2	127,188	58.61%	2.88%	62.59%
0.85	81	3	86,783	71.0%	5.81%	75.42%
0.85	81	4	42,996	84.17%	9.49%	88.21%
0.85	81	5	16,780	91.54%	10.81%	94.16%
0.85	81	6	5,686	94.32%	9.48%	94.94%
0.85	81	7	1,498	95.21%	6.74%	92.41%
0.85	81	8	395	95.4%	3.88%	84.79%
0.85	81	9	106	95.45%	2.2%	77.33%
0.85	81	10	35	95.46%	1.23%	63.26%
0.85	81	11	8	95.46%	0.5%	55.0%
0.85	81	12	3	95.46%	0.43%	47.93%

FIGURA 14. Tabla de similitud. Equivalente a la figura 10, para distintos valores de  $p$ . "Atributos" es la cantidad de datos preservados despues de la transformación. No incluye umbrales con 0 conexiones. Continua en la figura 15.

p	atributos	u	conexiones	coincidentes	adyacencia estirada	autovalores
0.9	110	0	298,805	7.6%	1.25%	48.01%
0.9	110	1	213,934	32.94%	2.31%	60.94%
0.9	110	2	127,396	58.56%	2.92%	62.97%
0.9	110	3	86,886	70.98%	5.85%	75.53%
0.9	110	4	43,122	84.15%	9.59%	88.34%
0.9	110	5	16,901	91.52%	10.96%	94.34%
0.9	110	6	5,773	94.31%	9.78%	95.2%
0.9	110	7	1,546	95.21%	7.03%	92.86%
0.9	110	8	416	95.4%	4.12%	85.64%
0.9	110	9	110	95.45%	2.23%	77.06%
0.9	110	10	37	95.46%	1.32%	63.46%
0.9	110	11	8	95.46%	0.5%	55.0%
0.9	110	12	3	95.46%	0.43%	47.93%
0.95	157	0	298,831	7.6%	1.26%	48.1%
0.95	157	1	213,950	32.93%	2.33%	61.01%
0.95	157	2	127,428	58.56%	2.94%	63.04%
0.95	157	3	86,902	70.98%	5.87%	75.55%
0.95	157	4	43,155	84.15%	9.62%	88.38%
0.95	157	5	16,900	91.53%	10.99%	94.38%
0.95	157	6	5,770	94.31%	9.8%	95.26%
0.95	157	7	1,544	95.21%	6.99%	92.9%
0.95	157	8	412	95.4%	4.27%	85.79%
0.95	157	9	114	95.45%	2.41%	78.33%
0.95	157	10	36	95.46%	1.35%	63.31%
0.95	157	11	9	95.46%	0.75%	56.17%
0.95	157	12	2	95.46%	0.06%	46.19%
0.99	237	0	298,804	7.6%	1.26%	48.03%
0.99	237	1	213,910	32.95%	2.33%	60.88%
0.99	237	2	127,386	58.57%	2.94%	62.99%
0.99	237	3	86,858	70.99%	5.86%	75.5%
0.99	237	4	43,097	84.16%	9.61%	88.3%
0.99	237	5	16,861	91.54%	10.99%	94.32%
0.99	237	6	5,730	94.32%	9.78%	95.08%
0.99	237	7	1,524	95.22%	6.99%	92.72%
0.99	237	8	406	95.4%	4.19%	85.64%
0.99	237	9	109	95.45%	2.32%	77.83%
0.99	237	10	36	95.46%	1.35%	63.31%
0.99	237	11	8	95.46%	0.5%	55.0%
0.99	237	12	2	95.46%	0.06%	46.19%

FIGURA 15. Continuación de la figura 14.

## REFERENCIAS

- [1] Gilles Brassard and Paul Bratley. *Fundamentals of Algorithmics*. 1995.
- [2] Richard L Burden and J Douglas Faires. *Numerical Analysis*. 2000.
- [3] L Ford and D Fulkerson. *Flows in networks*. Princeton University Press, 1962.
- [4] Jure Leskovec and Julian Mcauley. Learning to discover social circles in ego networks. *Advances in neural information processing systems*, 25, 2012.
- [5] Mark EJ Newman. The mathematics of networks. *The new palgrave encyclopedia of economics*, 2(2008):1–12, 2008.
- [6] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.